Buffer Sizing for TCP Flows in 802.11e WLANs

Tianji Li, Douglas J. Leith
Hamilton Institute, National University of Ireland Maynooth, Ireland
Email: {tianji.li, doug.leith}@nuim.ie

***Abstract*— We consider the task of sizing buffers for TCP flows in 802.11e WLANs. A number of fundamental new issues arise compared to wired networks. These include that the mean service rate is dependent on the level of channel contention and packet inter-service times vary stochastically due to the random nature of CSMA/CA operation. We find that these considerations lead naturally to a requirement for adaptation of buffer sizes in response to changing network conditions.**

***Index Terms*— WLAN, 802.11, 802.11e, TCP, Buffer sizing**

I. Introduction

In this paper we consider the task of sizing buffers for TCP flows in 802.11e WLANs. We focus on the typical deployment scenario where an infrastructure mode WLAN is configured with the Access Point (AP) acting as a wireless router between the WLAN and the Internet. TCP traffic is of particular importance in such WLANs as it currently carries the great majority (more than 90% [8]) of network traffic. Effects of buffer related issues in WLANs have received little attention in the literature. Exceptions include [5] which shows that appropriate buffer sizing can restore TCP upload/download unfairness, and [7] in which TCP performance with fixed AP buffer sizes and 802.11e is investigated. The present paper, which extends our previous work on buffer sizing for voice traffic [4], is to our knowledge the first work focussing on how to tune buffer sizes for TCP traffic in 802.11e WLANs .

Router buffers are traditionally sized with two primary objectives in mind.

(i) *Accommodating short-term packet bursts.* Due to the nature of TCP, internet traffic tends to be bursty. Should too many packets arrive in a sufficiently short interval of time, then a router may lack the capacity to process all of the packets immediately. The first job of the router buffer is to mitigate packet losses due to bursts by accommodating these packets in a buffer until they can be serviced.

(ii) *Ensuring AIMD throughput efficiency.* The AIMD congestion control algorithm used by TCP reduces the number of packets in flight by half on detecting network congestion. If router buffers are too small, this backoff action will cause them to empty with a corresponding reduction in link utilisation.

The classical rule of thumb is to provision buffers to be equal to the *bandwidth* of the link multiplied by the average *delay* (round trip time or RTT) of the flows utilising this link: the *Bandwidth-Delay Product* (BDP).

This work is supported by Science Foundation Ireland Grant 03/IN3/I396.

A number of fundamental new issues arise in 802.11e WLANs. Firstly, the mean service rate at a wireless station is strongly dependent on the level of channel contention and thus on the number of active stations and their load. Secondly, even when the network load is fixed, the packet inter-service times at a station are not fixed but vary stochastically due to the random nature of the CSMA/CA operation. These facts affect statistical multiplexing and buffer backlog behaviour, and thus the choice of buffer sizes.

In this paper we study the impact of these differences and find that they lead naturally to a requirement for adaptation of buffer size in response to changing network conditions. We propose an adaptive algorithm for 802.11e WLANs[1] and demonstrate its efficacy via simulations.

II. Network Setup

We consider scenarios where the Access Point (AP) acts as a wireless router between the WLAN and the wired Internet. Upload flows are from wireless stations in the WLAN to server(s) in the Internet, while downloads are from wired server(s) to stations in the WLAN. At the MAC layer, we use IEEE 802.11g parameters as shown in Table I. The bandwidth between the AP and server(s) is 100 Mbps. TCP Reno with SACK is used.

We note that in WLANs, TCP ACK packets without any prioritisation can be easily queued/dropped due to the fact that the basic 802.11 DCF ensures that stations win a roughly equal number of transmission opportunities. For example consider n stations each carrying one TCP upload flow. The TCP ACKs are transmitted by the AP. While the data packets for the n flows have an aggregate $n/(n+1)$ share of the transmission opportunities the TCP ACKs for the n flows have only a

[1]802.11e has been approved as an IEEE standard and much of the 802.11e functionality is already available in WLAN devices. The proposed algorithm however is also applicable to legacy 802.11 DCF although without TCP ACK prioritisation fairness between competing TCP flows can not be guaranteed.

T_{SIFS} (μs)	10
Idle slot duration (μs)	9
Retry limit	11
Packet size (bytes)	1000
PHY data rate (Mbps)	54
PHY basic rate (Mbps)	6
PLCP rate (Mbps)	6

TABLE I
MAC AND PHY PARAMETERS USED IN THIS PAPER.

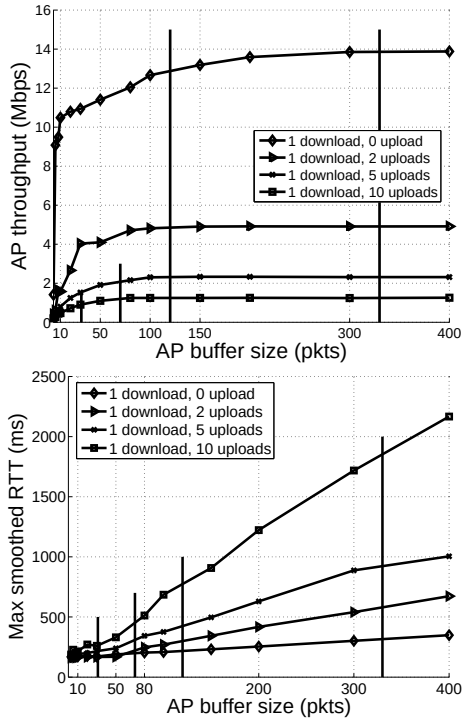


Fig. 1. Performance with fixed buffer sizes. Data shown for 1 download flow and 0, 2, 5, 10 uploads – corresponding BDP values marked by vertical lines. One upload flow per wireless station. Max smoothed RTT denotes the maximum TCP $srtt$ value observed. Wired backhaul link bandwidth 100Mbps, RTT 200ms.

$1/(n+1)$ share. Issues of this sort are known to degrade TCP performance significantly as queuing and dropping of TCP ACKs disrupt the TCP ACK clocking mechanism. Following [3], we address this problem using 802.11e. At the AP and each station we treat TCP ACKs as a separate traffic class, collecting them into a queue which is assigned high priority via $CW_{min} = 3$, $CW_{max} = 7$, $AIFS = 2$. TCP data packets are transmitted by another queue with parameters $CW_{min} = 31$, $CW_{max} = 1023$ and $AIFS = 6$. This makes use of 2 out of the 4 available queues in 802.11e.

III. Performance with Fixed Buffers

In contrast to wired networks, the mean service rate at a wireless station is not fixed but instead depends upon the level of channel contention and the network load. This is illustrated in Fig. 1 where the throughput and delay of a download flow are plotted as a function of the AP buffer size when the number of competing upload flows (with one upload flow per wireless station) is varied. Similarly to wired networks, the throughput always increases monotonically with the buffer size, reaching a maximum above a threshold buffer size. It can also be seen that the download throughput falls as the number of competing uploads increases. The variation in throughput can be substantial, e.g., the maximum throughput changes from 14Mbps to 1.25Mbps as the number of competing uploads changes from 0 to 10. As a result, the BDP – marked by vertical lines in Fig. 1 – also varies significantly and this is reflected in buffering requirements. For example, it can be seen from Fig. 1 that with no competing uploads the threshold buffer size above which the AP achieves maximum throughput is 338 packets, while for 10 competing uploads this buffer size falls to approximately 70 packets.

In view of this behaviour, one possible approach is to size buffers based on worst case conditions, i.e., based on the conditions requiring the largest buffering to achieve high throughput. However, while ensuring high throughput, this comes at the cost of high latency. For example, it can be seen from Fig. 1 that when a fixed buffer size of 338 packets is used (which in this example ensures maximum throughput regardless of the number of contending stations), the round-trip latency experienced by the download flow is about 300ms with no uploads but rises to around 2s with 10 contending upload stations. This occurs because TCP's congestion control algorithm probes for bandwidth until packet loss occurs and so download flows will tend to fill buffers with any sizes. Moreover, the mean queueing delay of a buffer of size Q with mean service rate B is Q/B. Hence, the queueing delay at the AP depends on the service rate, which in turn depends on the number of contending wireless stations and their offered load. For a fixed-size buffer, a decrease in the service rate of a factor b.

Conversely, sizing the buffer to achieve lower latency across all network conditions comes at the cost of reduced throughput, e.g., a buffer size of 30 packets ensures latency of 200-300ms for up to 10 contending upload stations but when there are no contending uploads the throughput of a download flow is only about 75% of the maximum achievable.

In addition to variations in the mean service rate, we also note that the random nature of 802.11 operations leads to short time-scale stochastic fluctuations in service rate. This is fundamentally different from wired networks and directly impacts buffering behaviour. Stochastic fluctuations in service rate can lead to early queue overflow and reduced link utilisation. For example, from Fig. 1 with 10 uploads the maximum download throughput is 1.25Mbps, yielding a BDP of 31 packets. However, it can be seen that at this buffer size the achieved download throughput is only about 60% of the maximum – a buffer size of at least 70 packets is required to achieve 100% throughput. The stochastic fluctuations in service rate lead to a requirement to increase the buffer size *above* the BDP in order to accommodate the impact of these fluctuations. The amount of over-provisioning required may be bounded using statistical arguments, but we do not pursue this further here due to space constraints. We note, however, that a simple but effective approach is to over-provision by a fixed number of packets above the BDP. For example, from Fig. 1 we can see that over-provisioning by 40 packets is sufficient for the range of conditions considered, and we find that this approach works more generally.

IV. Adaptive Buffer Sizing

Motivated by the foregoing observations and the difficulty of selecting a fixed buffer size suited to a range of network

conditions, we consider the use of an adaptive buffer sizing strategy. We note that a wireless station can readily measure its own service rate by observation of the inter-service time, i.e., the time between packets arriving at the head of the network interface queue t_s and being successfully transmitted t_e (which is indicated by receiving correctly the corresponding MAC ACK.). Note that this measurement can be readily implemented in real devices and incurs minor computation burden. Let $T_{serv}(t)$ be the inter-service time at time t, we use exponential smoothing to calculate the mean inter-service time $T_{serv} = \alpha T_{serv} + (1-\alpha)(t_e - t_s)$ where $\alpha = 0.999$ as per [2].

Using this measurement we propose the following adaptive strategy. Let T be the target queueing delay[2]. We then select buffer size Q according to $Q = min(T/T_{serv}, Q_{max})$ where Q_{max} is set to be 400 packets[3]. This will decrease the buffer size when the service rate falls and increase the buffer size when the service rate rises, so as to maintain an approximately constant queueing delay of T seconds. This effectively regulates the buffer size to remain equal to the BDP as the mean service rate varies.

To account for the impact of the stochastic nature of the service rate on buffer size requirements (see comments at the end of the previous section), we modify this update rule to $Q = min(T/T_{serv}+a, Q_{max})$ where a is an over-provisioning amount to accommodate short-term fluctuations in service rate. Based on the measurements in Fig. 1 and others, we have found that a value of a equal to 40 packets works well across a wide range of network conditions.

The effectiveness of this simple adaptive algorithm is illustrated in Fig. 2. Here we plot the throughput percentage[4] and smoothed RTT of download flows as the number of download and upload flows is varied. It can be seen that the adaptive algorithm maintains high throughput efficiency across the entire range of operating conditions. This is achieved while maintaining the latency approximately constant at around 400ms – the latency rises slightly with the number of uploads due to the over-provisioning a to accommodate stochastic fluctuations in service rate.

Fig. 3 demonstrates the ability of the adaptive algorithm to respond quickly to changing network conditions. At time 200s the number of uploads is increased from 0 to 10. It can be seen that the buffer size quickly adapts to the changed conditions while maintaining high throughput efficiency.

V. Conclusions and Future Work

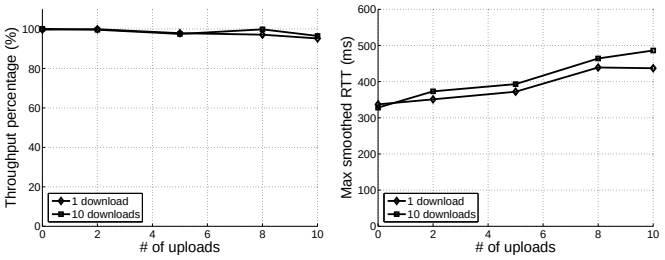
We consider the task of sizing buffers for TCP flows in 802.11e WLANs. A number of fundamental new issues arise compared to wired networks, including the fact that the mean service rate is dependent on the level of channel contention and packet inter-service times vary stochastically due to the random nature of CSMA/CA operation. Motivated by these observations we propose an adaptive buffer sizing algorithm which emulates the classical BDP rule and demonstrate its efficacy via simulations.

Buffer sizing while rate adaptation is enabled is left as future work, although we believe that the proposed algorithm will work. Future work also include consideration of the possibility of reducing buffer sizes when multiplexing occurs [1].

[2]We may select T based on the average RTT of the flows sharing the queue, but here we simply use a fixed value of 200ms since this is an approximate upper bound on the RTT of the majority of the current Internet flows.

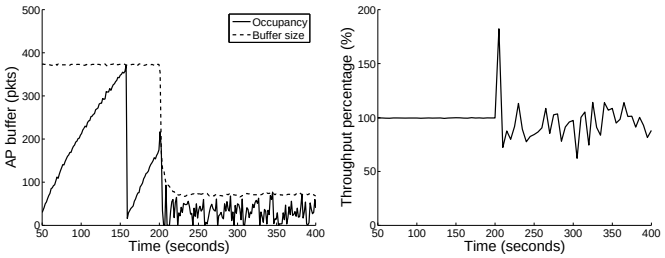
[3]We select $Q_{max} = 400$ packets as it is the default buffer size used by the popular Atheros chip sets. For future very high-speed WLANs such as 802.11n in which MAC layer throughput may reach 100 Mbps, a larger Q_{max} should be used.

[4]Throughput percentage is the ratio of throughput achieved by our algorithm to that with a 400-packet buffer.



(a) Throughput percentage (b) Max smoothed RTT

Fig. 2. Performance vs number of upload flows. Data is shown for 1, 10 download flows. Wired backhaul link bandwidth 100Mbps, RTT 200ms.



(a) AP buffer (b) Throughput percentage

Fig. 3. Convergence rate while adapting to changing network conditions. One download, at time 200s the number of uploads is increased from 0 to 10.

References

[1] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing router buffers," *Proc. ACM SIGCOMM*, 2004.

[2] C. Chatfield, *The Analysis of Time Series, An Introduction*, CRC Press 2004.

[3] D. J. Leith, P. Clifford, D. Malone, and A. Ng, "TCP Fairness in 802.11e WLANs," *IEEE Communications Letters*, vol. 9, no. 11, pp 964–966, 2005.

[4] D. Malone, P. Clifford, and D. J. Leith, "On Buffer Sizing for Voice in 802.11 WLANs," *IEEE Communications Letters*, vol. 10, no. 10, pp 701–703, 2006.

[5] S. Pilosof, et. al., "Understanding TCP fairness over Wireless LAN," in *Proc. IEEE Infocom* 2003.

[6] R. Stanojevic, C. M. Kellett, and R. N. Shorten, "Adaptive Tuning of Drop-Tail Buffers for Reducing Queueing Delays," *IEEE Communications Letters*, vol. 10, no. 7, pp 570–572, 2006.

[7] M. Thottan, and M. C. Weigle, "Impact of 802.11e EDCA on mixed TCP-based applications," in *Proc. IEEE WICON* 2006.

[8] Z. Zhao, S. Darbha, and A. L. N. Reddy, "A method for estimating the proportion of nonresponsive traffic at a router," *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 708–718, 2004.